(19) JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: 07152640 A

(43) Date of publication of application: 16 . 06 . 95

(51) Int. Cl G06F 12/06
G06F 15/163

(21) Application number: 05296300

(22) Date of filing: 26 . 11 . 93

(71) Applicant: HITACHI LTD

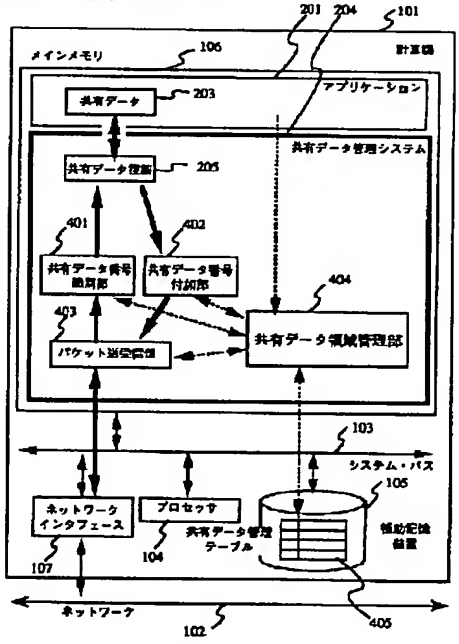
(72) Inventor: YAMAUCHI MASAHIKO
YOSHIZAWA SATOSHI
MURAYAMA HIDEKI
HAYASHI TAKEHISA
KITO AKIRA

(54) DECENTRALIZED COMMON MEMORY SYSTEM

(57) Abstract:

PURPOSE: To decrease the amount of transferred data and the frequency of transfer and to effectively utilize network resources by obtaining information regarding common data such as the timing, etc., of the guarantee for the consistency of common data areas and common data as to the common data that application software uses as if one data were shared for the application software.

CONSTITUTION: A common data management system 204 consists of a common data number discrimination part 401, a common data number addition part 402, a packet transmission and reception part 403, and a common data area management part 404, and hold respective constituent elements of a common data management table 405. When the application software 201 declares data (common unit) to be shared and mapping to a memory to the common data area management part 404, the correspondence between the common data and memory is stored in the common data management table 405. A computer which indicates common data modification and a computer which receives the common data modification guarantee the consistency of the common data by using the respective constituent elements.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平7－152640

(43)公開日　平成７年(1995)６月16日

(51)Int.Cl.⁶	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　12/06	５３０　Ｆ	9366－5B		
15/163				
		8219－5L	Ｇ０６Ｆ　15/16　３２０　Ｍ	

審査請求　未請求　請求項の数13　ＯＬ　（全 19 頁）

(21)出願番号　特願平5－296300

(22)出願日　平成５年(1993)11月26日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　山内　雅彦
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(72)発明者　吉沢　聡
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(72)発明者　村山　秀樹
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(74)代理人　弁理士　小川　勝男

最終頁に続く

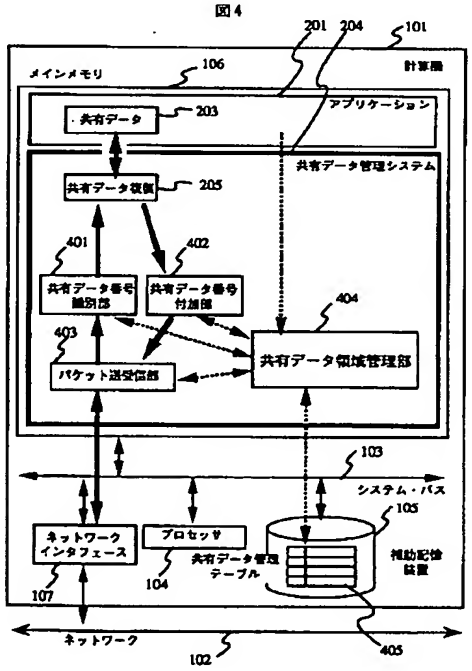
(54)【発明の名称】　分散共有メモリ方式

(57)【要約】

【目的】　アプリケーションに対してあたかも一つのデータを共有しているように、アプリケーションが使用する共有データに関して、共有するデータ領域や、共有するデータ一貫性保証のタイミング等の共有データに関する情報を得て、データ転送量や転送回数を減らし、ネットワーク資源を有効に活用する。

【構成】　共有データ管理システム204は、共有データ番号識別部401、共有データ番号付加部402、パケット送受信部403、共有データ領域管理部404から構成され、共有データ管理テーブル405の各構成要素を保持する。アプリケーション201が共有するデータ（共有単位）とメモリへの写像を共有データ領域管理部404に対して宣言すると、共有データとメモリとの対応を共有データ管理テーブル405に格納する。共有データ変更を指示した計算機と共有データ変更を受け取る計算機では、上記各構成要素を用いて共有データの一貫性を保証する。

図４



1

【特許請求の範囲】

【請求項１】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ方式において、

計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共有単位で指定し、指定された共有単位にて共有データを対応付けて管理する共有データ管理手段を上記複数の計算機の少なくともいずれか一つに設けたことを特徴とする分散共有メモリ方式。

【請求項２】請求項１に記載の分散共有メモリ方式において、上記共有データの複製を格納する手段を設け、計算機のアプリケーションは、上記複製に対してＲｅａｄ／Ｗｒｉｔｅを行うことを特徴とする分散共有メモリ方式。

【請求項３】請求項１または２に記載の分散共有メモリ方式において、各計算機間で共有する領域間の一貫性保証を、任意のアドレス及びサイズよりなる共有単位で指定することを特徴とする分散共有メモリ方式。

【請求項４】請求項１または２に記載の分散共有メモリ方式において、各計算機間で共有する領域の任意の部位を一次元配列への写像として指定し、その一次元配列を共有単位として指定することを特徴とする分散共有メモリ方式。

【請求項５】請求項４に記載の分散共有メモリ方式において、上記一次元配列を格納する領域を、共有単位が指定された際に、メモリ上で確保することを特徴とする分散共有メモリ方式。

【請求項６】請求項３に記載の分散共有メモリ方式において、各計算機間で共有する複数の共有単位について、共有単位間での階層関係を定義し、上記一貫性保証を指定した共有単位に加え、その下位階層として定義された共有単位についても一貫性保証を一括して指定することを特徴とする分散共有メモリ方式。

【請求項７】請求項６に記載の分散共有メモリ方式において、上記一貫性保証を行なう際に、指定された共有単位の下位階層として定義された共有単位から、任意の下位階層共有単位を選択して、一貫性保証を一括して行なうことを特徴とする分散共有メモリ方式。

【請求項８】請求項３に記載の分散共有メモリ方式において、各計算機間でデータを交換する手段として、上記一貫性保証した共有単位のデータ交換手段と共に、他のデータ交換手段を備え、交換するデータのメモリ上での配置に応じてデータ交換手段を選択することを特徴とする分散共有メモリ方式。

【請求項９】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ方式において、

計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共

2

有単位で指定し、指定された共有単位にて共有データを対応付けて管理するための共有データ管理テーブルと、

上記共有データ管理テーブルを管理する共有データ領域管理手段と、

他の計算機とのパケットによる上記共有データを送受信するパケット送受信手段と、

上記パケット送受信手段を介して送信するパケットを生成するために、共有データを格納するメモリ領域を調べ、そのメモリ領域に対して共有データ番号を付与する共有データ番号付与手段と、

上記パケット送受信手段を介して受信したパケット中の共有データ番号を識別し、共有データ番号に対応するメモリ領域を上記共有データ管理テーブルを用いて調べ、対応するメモリ領域に共有データを書き込む共有データ番号識別手段と、

計算機のアプリケーションからＲｅａｄ／Ｗｒｉｔｅを行うために共有データの複製を格納する手段とを、

上記複数の計算機のうち少なくともいずれか一つに設けたことを特徴とする分散共有メモリ方式。

【請求項１０】請求項９に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、共有データ番号を格納する共有データ番号エントリと、共有データを管理している計算機かどうかを示す管理ノードフラグエントリと、共有データの開始アドレスを示す共有データアドレスエントリと、共有データの複製の開始アドレスを示す共有データ複製アドレスエントリと、共有データの大きさを示す共有データサイズエントリと、共有データを保持している計算機を示す計算機リストエントリを有することを特徴とする分散共有メモリ方式。

【請求項１１】請求項１０に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、アプリケーションが使用する共有データが連続領域か否かを示す分割フラグエントリと、幾つの分割されたデータから共有データが構成されているかを示す共有データ要素数エントリと、分割されたデータがどのくらいの間隔で並んでいるかを示す共有データオフセットエントリを有することを特徴とする分散共有メモリ方式。

【請求項１２】請求項１１に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、共有データが構造体である場合に、構造体の定義を格納する構造体定義エントリと、構造体のデータ長を格納する構造体サイズエントリと、複数の共有データに階層関係があることを示す階層要素フラグエントリと、階層要素の共有データ番号を格納する階層要素共有データ番号エントリを有することを特徴とする分散共有メモリ方式。

【請求項１３】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ処理方法において、

計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共

有単位で指定し、
指定された共有単位にて共有データを対応付けて管理する管理計算機を決定して、上記管理計算機に対して共有データの使用を要求し、
上記管理計算機では、指定された共有データが使用されているか否かを確認し、
使用されていない場合には、指定された共有データに関する情報を上記管理計算機に新規に登録し、
使用されている場合には、新規に共有データの使用を指定した計算機に関する情報を上記管理計算機に登録して、その情報を他の計算機に通知し、
共有データの使用を指定した計算機では、各計算機のメモリ上に分散した共有データを獲得して、共有データの複製を作成する、
ことを特徴とする分散共有メモリ処理方法。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、ネットワークで接続された複数の計算機システムで構成されるネットワーク計算機システムに於いて、物理的に分散配置されたメモリ上のデータを、別計算機上で実行中の複数アプリケーション間で共有するための分散共有メモリ技術に係わり、特にネットワーク資源を有効利用し、その結果、分散共有メモリへのアクセス性能を向上させる分散共有メモリ方式に関する。

【０００２】

【従来の技術】計算機に於いて、計算機内部でアプリケーションを実行する仮想的な実体をプロセスと呼ぶ。プロセスを複数生成し、単一ＣＰＵの演算資源を一定時間毎に切り換えて各プロセスに割り当てることで、ＣＰＵ台数よりも多いプロセスを並行に動作させることができる。これを並行処理と呼ぶ。一方、複数のＣＰＵを使用して、複数のプロセスを同時に動作させる処理を並列処理と呼ぶ。

【０００３】従来、単一ＣＰＵ構成の計算機に於いて並行処理を行なう時には、プロセス間でのデータ交換の方法として共有メモリを使用することができた。これには例えば、「ＵＮＩＸシステムコール・プログラミング、Ｍａｒｃ　Ｊ．Ｒｏｃｈｋｉｎｄ著、福崎俊博訳、アスキー出版、１９８７」の２９９－３１５頁に示されたシステムＶの共有メモリがある。

【０００４】並行処理では物理的にプロセスが同時に実行されないために、複数プロセスによって構成されるアプリケーションの実行性能の向上は望めなかった。そこで、ネットワーク計算機システムでは、複数の計算機をネットワークで接続し、複数のＣＰＵを同時に利用してアプリケーションを並列処理することによって、アプリケーション実行性能の向上を図る。

【０００５】ネットワーク計算機システムに於ける共有メモリでは、メモリが物理的に分散しているので、各計算機のメモリ上に格納されたデータ間の一貫性を保証し、あたかも一つのデータを共有しているかの様に、アプリケーションに見せる必要がある。この技術を分散共有メモリと呼ぶ。

【０００６】仮想記憶機構をページング機構で実現する計算機では、そのページング機構を拡張して分散共有メモリを実現することができる。即ち、通常ではメモリアクセスの際、アクセスしようとしたデータがメモリ上に存在しないとページフォールトを発生し、補助記憶装置からデータが存在するページをメモリ上にロードする。分散共有メモリでは、ページフォールト発生の際、補助記憶装置からページを取得する代わりに、他計算機からネットワークを介して自分の計算機のメモリ上にロードすることで実現される。

【０００７】分散共有メモリの従来技術の例としては、以下のＩｖｙシステム（米・エール大学）、及びＤＡＳＨシステム（米・スタンフォード大学）を挙げることができる。

【０００８】Ｉｖｙシステムでは、各ページに所有者（オーナ）が存在し、所有者である計算機が、そのページの複製を保持する計算機の名称を管理する。また所有者のみがそのページへ書き込みを行なうことを許されている。ある計算機で読み出しフォールトが発生した時には、所有者からページの複製が送られる。また、書き込みフォールトが発生した時には、書き込みフォールトを発生した計算機が新しい所有者となり、他の計算機上に存在するページの複製は無効化される。

【０００９】ＤＡＳＨシステムでは、１６バイトのデータを単位として、Ｉｖｙシステムと同様な制御を行なっている。

【００１０】これらの従来技術の詳細については、「分散オペレーティングシステム　ＵＮＩＸの次にくるもの、前川守・所真理雄・清水謙多郎編、共立出版、１９９１」の１２４－１４１頁に示されている。

【００１１】

【発明が解決しようとする課題】分散共有メモリではメモリ内容の一貫性を保つためにネットワークで接続された計算機間でメモリ内容の交換が行なわれる。この時、従来の分散共有メモリ管理方式で課題となるのはデータ転送サイズとデータ転送タイミングである。

【００１２】Ｉｖｙシステムについては以下の様な課題がある。第１にデータ転送サイズに関して、メモリ内容交換のために計算機間を流れるデータ転送サイズがページであるため、ページサイズ（例えば、４キロバイト）と比較して小さいデータ（例えば、８バイト）を計算機間で共有している場合、計算機間を流れるデータ転送量に無駄が発生する。このため、実質的なネットワーク資源の利用率が下がるという課題がある。

【００１３】また、計算機Ａと計算機Ｂの各計算機でアクセス頻度が高い２つの異なるデータが同一ページ上に

存在する場合に、計算機Ａと計算機Ｂが交互に同じページに対するメモリ書き込みを行なうと、ページが計算機Ａと計算機Ｂの間を移動する。このため、各計算機がメモリアクセスを行なう度に、ネットワークを介した計算機間でのページ転送が発生し、メモリアクセスの性能が下がるという課題がある。

【００１４】第２にデータ転送のタイミングに関して、データ転送はメモリへの読み出しや書き込み時に自計算機内に共有データの存在するページがない時にページフォルトを発生して行なっていた。

【００１５】そのため計算機Ａと計算機Ｂが同じ共有データに連続して書き込みを行うと、最悪の場合、各計算機がメモリ書き込みを行なう度にページが計算機間を移動する。ネットワークを介して計算機間の転送を行なうコストは、自分の計算機内のメモリアクセス速度と比較して大きいため、共有データへのメモリアクセスの性能が下がるという課題がある。

【００１６】一方ＤＡＳＨシステムは、Ｉｖｙシステムの第１の課題の解決を狙ったシステムである。即ち、ページ単位でメモリ内容を共有しているために発生する無駄なデータ転送を無くすために共有の単位を小さくした。その結果、無駄なデータ転送はなくなったが、メモリの広範囲に対してアクセスを行なうとアクセスフォールトが頻繁に発生することとなった。アクセスフォールトが頻繁に発生すると、ネットワークを介して行なう計算機間でのデータ転送時間が顕著になる。ＤＡＳＨシステムでは、これに対して小さいデータの転送時間を短くすることを目的として、特殊化した専用ネットワークを提案している。しかし、この専用ネットワークは汎用ネットワーク・アーキテクチャとは異なるためコストが高いという問題がある。

【００１７】更にＩｖｙシステムの第２の課題である、複数計算機が同じ共有データに対して連続してアクセスした場合の性能低下については、ＤＡＳＨシステムに於いても発生し解決されていない。

【００１８】上記課題に起因して、従来の分散共有メモリ方式を、典型的な並列処理手法である領域分割型並列処理に適用した場合、ネットワークを介した通信が頻繁に発生し、並列化による処理の高速化率で良好な値を得ることができない。更にポインタ変数を要素とするデータ構造体を複数計算機間で共有する場合等、共有の対象となるデータがメモリ上で散在して格納されることがあり、この様な場合にもネットワークを介した通信が頻繁に発生する。

【００１９】本発明の目的は、上記課題を解決し、ネットワーク資源の有効活用、及びアクセス性能の高い分散共有メモリ方式を提供することにある。

【００２０】

【課題を解決するための手段】先ず、第１の課題を解決するために、アプリケーションで共有するデータとメモリへの写像を設定可能とし、設定した共有単位を記憶しておくための共有データ管理テーブルを有し、その共有データ管理テーブルを管理するための共有データ管理手段を設ける。

【００２１】具体的には、アプリケーションが２次元配列の様なデータを２分割し、分割した境界部分のデータを共有する場合には、アプリケーションからその境界部分のデータを共有単位として設定する。共有単位は、メモリ連続領域、メモリ不連続領域（データサイズ一定、データの間隔一定）、またはそれらの組合わせによって指定する。

【００２２】次に第２の課題を解決するために、複数の計算機上の共有データの一貫性を保証するタイミングをアプリケーションが仮想的な共有空間からの（１）最新データの獲得指示、共有空間への（２）最新データの反映指示という２つの情報に基づいて決定し、一貫性を保証する単位として設定された共有単位を使用する。

【００２３】具体的には、全ての計算機に共有データを持たせ、共有データを変更する計算機に於ける（２）最新データの反映指示のタイミングで全ての共有データの更新を行なう方法、または一つの計算機に共有データを持たせて、共有データを獲得しようとした計算機に於ける（１）最新データ獲得指示のタイミングで共有データを保持している計算機から共有データをネットワークを介して獲得する方法、更に、共有データの信頼性を高めるために、これら２つの方法を組み合わせて一つ以上の計算機で共有データを保持し、危険を分散してもよい。

【００２４】

【作用】アプリケーション実行時に、予め共有単位を設定することによって無駄なデータ転送を防ぎ、ネットワーク資源を有効に活用することができる。また、物理的に分散しているデータの一貫性をアプリケーションが必要な時のみ保証することによって、ネットワークを介した計算機間の転送を最小現に抑さえてネットワーク資源を有効活用できる。その結果、共有データへのアクセス性能を向上できる。

【００２５】

【実施例】図１は、本発明に於けるネットワーク計算機システム１００の一実施例を示すブロック図である。

【００２６】本図に於いて、ネットワーク計算機システム１００は、複数の計算機１０１をネットワーク１０２で接続して構成される。計算機１０１に於いて、１０３は計算機を構成する各ブロック間のデータ伝送路であるシステム・バス、１０４はプログラムを実行するプロセッサ、１０５はプログラムやデータを格納する磁気ディスクなどの補助記憶装置、１０６はプログラム実行中に該プログラムやデータを格納するためのメインメモリ、１０７はネットワーク１０２との入出力を制御するためのネットワーク・インタフェースである。本発明による共有データ管理方式は、例えば計算機１０１（１）上の

アプリケーションと計算機１０１（２）上のアプリケーションが並列実行する時に、データを共有する際に適用される。

【００２７】図２は、領域分割型並列処理に対して、本発明による分散共有メモリを適用する場合の一実施例を示す図である。

【００２８】ここで領域分割型並列処理とは、図２（ａ）に示す様なデータ配列を、図２（ｃ）、（ｄ）に示す様に分割して、それぞれを別々の計算機上で並列に処理する形式の並列処理のことである。流体解析問題や構造解析問題等、多くの科学技術計算には、領域分割型並列処理を適用可能である。

【００２９】領域分割型並列処理の一例として、二次元熱拡散解析問題の一解法を図２を用いて説明する。本解析手法では、解析対象となる領域を図２（ａ）に示す様なｎ×ｎ二次元メッシュ状に区切り、配列データ２０２としてメインメモリ１０６上に格納して、プロセッサ１０４が演算処理を行なう。この際、配列データ２０２の各要素には、各要素の温度値を格納する。演算処理は、図２（ａ）の二次元配列の全要素に対して、各要素の近傍要素を用いた計算を行ない、それを各要素の一定時間後の値として設定する処理を、予め定められた終了条件が満たされる迄、繰り返し行なう形態をとる。ここで各要素の近傍要素を用いた計算とは、例えば図２に示した様に、配列要素（ｍ，ｍ）の新たな値を計算する際に、その４近傍点、即ち（ｍ，ｍ－１），（ｍ＋１，ｍ），（ｍ，ｍ＋１），（ｍ－１，ｍ）と、（ｍ，ｍ）自体の値を用いて計算することである。本計算の一例は、配列要素（ｍ，ｍ）の値に対して、各近傍点の値と（ｍ，ｍ）との値の差異を先ず求め、それに熱伝導率に該当する係数を乗じた上で、（ｍ，ｍ）の値に加算するものである。

【００３０】図２（ｃ）、（ｄ）に、本解析手法を二台の計算機上で並列に処理する場合の領域分割を示す。上記の様に本解析では各配列要素の演算のために近傍点のみを用いるため、ｘ＝０からｘ＝（ｎ／２）－２迄の列２０２（１）、及びｘ＝（ｎ／２）＋１からｘ＝（ｎ－１）迄の列２０２（２）に関しては、各領域の演算を担当する計算機のみが読み書きできれば良く、各計算機のローカルメモリ１０６上に格納されていれば良い。それに対して、ｘ＝（ｎ／２）－１からｘ＝（ｎ／２）までの列２０３（１）及び（２）は、両計算機から読み書きできる必要があり、共有メモリ領域内に格納される必要がある。

【００３１】尚、上記の様な解析問題における典型的な問題規模、即ち図２（ａ）の配列の要素数は、数百行×数百列から数千行×数千列程度のオーダである。

【００３２】以下に説明する様に、本発明の分散共有メモリ方式では、計算機間でのデータ交換を、上記の様な領域分割型並列処理に対しても効率良く行なえる方式を提供する。

【００３３】図３は、同一のアプリケーションを実行する複数のプロセス間に於いてデータを共有する際の共有データ管理方式の一実施例を示すブロック図である。

【００３４】本図に於ては、計算機１０１（１）上のアプリケーション２０１（１）と、計算機１０１（２）上のアプリケーション２０１（２）とがデータを共有している。ネットワーク計算機システム１００を構成する各計算機１０１は物理的に共有したメモリを持っていないために、物理的に分散した共有データ２０３（１）と２０３（２）との一貫性をハードウェア的、ソフトウェア的、またはそれらの組み合わせによる手段で保つ必要がある。この一貫性制御によって、計算機１０１（１）上のアプリケーション２０１（１）と、計算機１０１（２）上のアプリケーション２０１（２）とが、あたかも一つのデータを共有している様に見せることができる。

【００３５】本実施例では、共有データの一貫性を保つ仮想的な空間を各計算機上に存在する共有データ管理システム２０４によって提供する。アプリケーション２０１が共有データに対してＲｅａｄ／Ｗｒｉｔｅを行う時には、共有データ管理システム２０４によって一貫性が保証された仮想的な空間との間で行なう。本実施例では具体的には、共有データ複製領域２０５に格納されたデータを読み書きして行なう。

【００３６】本実施例に於ては、全ての計算機１０１の共有データ管理システム２０４には、アプリケーション２０１で使用する共有データ２０３の複製２０５が存在する。この複製２０５は、アプリケーションが共有データを使用することを宣言した時に生成される。これはアプリケーション２０１から共有データ２０３を獲得する要求がきた時の処理遅延を最小限にするためである。共有データ管理システム２０４では、物理的に分散している共有データの複製２０５に対して一貫性を保証する。即ち、共有データ管理システムは、ある計算機で共有データの複製を変更した場合には、共有データの複製２０５を保持する全ての計算機１０１に対して変更を伝達する。

【００３７】また、共有データ管理システム２０４の必要とするメモリ量を節約するために、全ての計算機に共有データの複製を持たずに、アプリケーション２０１が最新の共有データ反映の指示を行った時に、その指示を行った計算機上に複製を作成する方式をとっても良い。同時に、共有データの複製２０５を保持する計算機の識別子を他の計算機に通知する。共有データの複製は、ネットワーク計算機システム内の一台、または一台以上の計算機が保持していても良い。

【００３８】図４は、本発明による共有データ管理方式の一実施例を示すブロック図である。

【００３９】共有データ管理システム２０４は、共有デ

ータ番号識別部４０１、共有データ番号付加部４０２、バケット送受信部４０３、共有データ領域管理部４０４から構成される。また、補助記憶装置１０５には、共有データ管理テーブル４０５を持つ。共有データ管理テーブル４０５は、予めメインメモリ１０６に展開しておくことによって、その参照、更新処理を高速化することができる。共有データ管理テーブル４０５の詳細については後述する。

【００４０】共有データ番号識別部４０１は、バケット送受信部４０３から送られてくるバケットの共有データ番号を識別する。共有データ番号に対応するメモリ領域を共有データ領域管理部４０４を利用して調べ、対応するメモリ領域に対してデータを書き込む処理を行なう。

【００４１】共有データ番号付加部４０２は、共有データ領域管理部４０４から共有データ番号とそれに対応するメモリ領域を受け取り、メモリ領域を読み込んで他の計算機に共有データの内容を伝達するバケットの生成を行なう。

【００４２】バケット送受信部４０３は、ネットワーク・インタフェース１０７から送られてくるバケットのコマンドを解析し、そのバケットを処理するブロックに振り分ける。また、共有データ番号付加部４０２で作成したバケットを他の計算機に伝達するためにネットワーク・インタフェース１０７を起動する処理を行なう。

【００４３】共有データ領域管理部４０４は、共有データ管理テーブル４０５の管理と共有データ管理システム２０４のブロックを制御する処理を行なう。また、アプリケーション２０１の共有データに対する処理依頼を受け取り、共有データ管理テーブル４０５に新しいエントリを追加したり、削除を行なう。更に、共有データの内容を他の計算機に伝達する様に共有データ番号付加部４０２やバケット送受信部４０５を制御する。逆に共有データの内容を他の計算機から受け取れる様に共有データ番号識別部４０１やバケット送受信部４０３を制御する。

【００４４】図中４０１から４０４に於いて行なわれる各処理は、ソフトウェア的、ハードウェア的、またはその組み合わせによって行なわれるものである。ソフトウェア的に行なわれる部分の処理に関しては、以下の様に実行される。即ち、各ソフトウェア処理の内容を記述したプログラムは補助記憶装置１０５に格納されており、システム・バス１０３に接続されたメインメモリ１０６に展開した上で、プロセッサ１０４によって実行される。

【００４５】図５は領域分割型並列処理を行う際に、配列データを分割した時の共有データのメモリ配置を示す一実施例である。

【００４６】本図に於ては簡単化のために、６行×６列の配列を列方向に２分割して、２台の計算機を利用して並列処理することを例に採る。実際にはより要素数の多い配列を、行方向、及び列方向に分割して、より多数の計算機に割り当てて並列処理を行なうことが想定されるが、本図を用いて説明する内容を直接適用することが可能である。

【００４７】配列の各要素は８バイト長の倍精度実数とする。分割した境界部分の１列を共有データ番号「０」を持つ共有データ５０１、共有データ番号「１」を持つ共有データ５０２として宣言する。５１１は、６行×３列の部分配列のメインメモリ上の配置を示している。５１１が示す様に共有データ５０１はメインメモリ上では不連続領域として配置されている。

【００４８】一方、本発明に於て共有データを宣言した時に作成される共有データ複製のメインメモリ上の配置の一実施例は、５１２に示す様に連続領域になっている。５１１に於ける共有データの配置と５１２に於ける共有データ複製との配置の対応を共有データ管理テーブル４０５に於いて管理している。領域５１１と５１２に格納されたデータの対応付けは、共有データ番号をキーとして行っている。この対応はアプリケーションが５１１に於ける共有データを使用すると宣言した時に共有データ管理テーブル４０５内に自動的に作成する。これによって、不連続な共有データを一つの識別子（共有データ番号）で指定でき、かつ不連続なデータを一つにまとめることによって無駄なメモリ領域を必要としない。

【００４９】図６は、本発明に於ける共有データ管理テーブル４０５の一実施例を示し、特に領域分割型並列処理に適したテーブル構造を示す図である。

【００５０】共有データ管理テーブル４０５は、共有データ番号エントリ６０１、共有データを管理している計算機かどうかを示す管理ノードフラグ・エントリ６０２、共有データの開始アドレスを示す共有データアドレス・エントリ６０３、共有データの複製の開始アドレスを示す共有データ複製アドレス・エントリ６０４、共有データの大きさを示す共有データサイズ・エントリ６０５、共有データを保持している計算機リスト・エントリ６０９からなる。

【００５１】また、アプリケーションが使用する共有データが連続領域か否かを示す分割フラグ・エントリ６０６、幾つの分割されたデータから共有データが構成されているかを示す共有データ要素数エントリ６０７、分割されたデータがどのくらいの間隔で並んでいるかを示す共有データオフセット・エントリ６０８を保持する。これによって、アプリケーションは連続したメモリ領域の共有データだけでなく、一定サイズのデータが一定間隔で並んだメモリ領域を共有データとして宣言することが可能である。

【００５２】更に計算機リスト・エントリ６０９によって、共有データ番号で示される共有データを保持する全ての計算機を特定することができる。共有データ変更を他の計算機に通知するために、計算機リスト・エントリ

６０９を利用してもよい。利用しない場合には、ネットワークに接続されている全ての計算機に通信を行なうブロードキャスト機能を使えばよい。但し、ここで要求されるブロードキャスト機能は、パケット消失が発生しない、各計算機で受信するパケットの順番が変わらないという信頼性を持つ必要がある。

【００５３】共有データの複製を全ての計算機が保持しない場合には、共有データの複製を保持している計算機の識別子を格納する項目を必要とする。

【００５４】図７は、複数計算期間で、ポインタ変数を含むデータ構造体を共有する際の共有データのメモリ配置を示す一実施例である。

【００５５】図７（ａ）にＣ言語でのデータ構造体の宣言の一例を示す。本例に於ては三番目の要素ｃが正数型のポインタ変数となっている。以下、本構造体を共有する場合を例に、本発明による一実施例を示す。

【００５６】図７（ａ）に示したデータ構造体を共有する場合、メモリ上では図７（ｂ）の５１１に示す様に配置される。本配置に於て、要素ｃの領域７０１に格納されるのはメモリ上のアドレス値であり、これによりポインタを介してデータ領域７０２を指し示す。データ構造体を複数計算機間で共有する場合、共有したいデータは要素ａ，ｂ，ｄ、及び領域７０２に格納されたデータであり、要素ｃに格納されたアドレス値自体は共有する必要がない。これは各計算機が独立したメインメモリ１０６を管理しており、各計算機上での領域５１１内の共有データ格納アドレスは必ずしも一致しないためである。

【００５７】一方、本発明に於て共有データを宣言した時に作成される共有データ複製のメインメモリ上の配置の一実施例を５１２に示す。５１２では、５１１で要素ｃを格納していた領域７０１に該当する７０３の領域には、当該データ格納領域の開始位置迄のオフセット値７０４を格納する。またオフセット値７０４で指定されるデータ格納領域の先頭７０５には、データ長７０６を格納し、それに連続してデータ７０２を格納する。これにより、ポインタ変数を要素とするデータ構造体を複数計算機間で共有する場合にも、共有の対象となるデータはメモリ上で連続した領域に格納することができる。

【００５８】５１１に於ける共有データの配置と５１２に於ける共有データ複製との配置の対応を共有データ管理テーブル４０５に於いて管理している。領域５１１と５１２に格納されたデータの対応付けは、対応付けは共有データ番号をキーとして行っている。この対応は、左記に図５で示したのと同様に、アプリケーションが５１１に於ける共有データを使用すると宣言した時に共有データ管理テーブル４０５内に自動的に作成する。これによって、不連続な共有データを一つの識別子（共有データ番号）で指定でき、複数計算機間で共有する場合にネットワークを介した通信の頻繁を減らすことができる。

【００５９】図８は、本発明に於ける共有データ管理テーブル４０５の他の実施例を示し、特にデータ構造体の共有、及び階層関係を持つ共有データの管理について示す図である。

【００６０】本実施例に於ては、図６に示した共有データ管理テーブル４０５に対して、６１０の構造体定義エントリ、及び６１１の構造体サイズを示すエントリを設けている。共有データ番号６０１に対応する共有データがデータ構造体の場合、構造体定義エントリ６１０にはデータ構造体の定義を格納し、その共有データ複製領域５１２上でのデータ長を６１１に格納する。本図に於ては、共有データ番号「１１」の共有データが、データ構造体であることを示している。

【００６１】共有データ管理テーブル４０５の構造体定義エントリ６１０に格納する構造体定義テーブルの一実施例を図９（ａ）に示す。構造体定義テーブル８０１は、当該データ構造体の要素の型情報を格納するエントリ８０２、当該要素のサイズを格納するエントリ８０３、当該要素の当該データ構造体内での開始位置を格納するエントリ８０４、当該要素がポインタ型の場合に、それが指し示すデータのサイズを格納するデータサイズ・エントリ８０５により構成される。

【００６２】更に図８に於て、階層要素フラグ・エントリ６１２、及び階層要素共有データ番号エントリ６１３を設けることにより、複数の共有データに対して階層関係を定義しておき、共有データの一貫性保証処理のための獲得処理等を下位階層として定義されたデータを含めて、一纏めで実行することが可能となる。例えば図８に於ては、共有データ番号「５」の共有データの下位階層データとして、共有データ番号「６」の共有データが定義されており、更に共有データ番号「６」の下位階層には何も定義されていない。この場合、例えば共有データ番号「５」の一貫性保証処理を指示すると、共有データ番号「６」の共有データに関しても同様の処理を実行する。また共有データ番号「６」の一貫性保証処理を指示した場合には、その共有データに対してのみ、同処理を実行する。

【００６３】尚、階層関係にある共有データは、共有データ複製領域５１２上で連続して配置しておくことによって、一貫性保証処理を実行する際に一括してネットワークにデータを送出することが可能となり、ネットワークの利用効率を向上することができる。

【００６４】また、図７のポインタ変数を含むデータ構造体を階層構造として管理することもできる。即ち、データ構造体の要素ａからｄの本体部分を一つの共有単位として扱い、その下位階層の共有単位として、要素ｃが指し示すデータ部分７０２を定義すれば良い。また複数のポインタ変数がデータ構造体内に定義されている場合には、例えば構造体定義に表れる順番で、下位階層の共有データ番号を階層要素共有データ番号エントリ６１３に登録すれば良い。

【００６５】ポインタ変数を含むデータ構造体を階層構造として管理する別の実施例を、図９（ｂ）に示す。構造体定義テーブル８０２は、８０１と同等の８０３から８０５のエントリと、当該要素がポインタ型の場合に、それが指し示すデータの共有データ番号を格納するエントリ８０６により構成される。

【００６６】図１０は、本発明による共有データ管理システム２０４が使用するパケット構造の一実施例を示す図である。

【００６７】パケット９０１は、パケットヘッダ９０２とデータ９０３より構成される。パケットヘッダ９０２は、送信先計算機９０４、送信元計算機９０５、共有データ番号９０６、共有データ管理コマンド９０７より構成される。例えば、共有データ番号１２を持つ共有データの変更を計算機１が計算機２に通知する場合には、送信先計算機９０４に「２」、送信元計算機９０５に１、共有データ番号９０６に１２、共有データ管理コマンド９０７に「共有データ変更通知」を示す識別子を、データ９０３に変更された共有データの内容を格納して、共有データを保持する計算機もしくは全ての計算機にパケットを送信する。

【００６８】図１１は、本発明による共有データ管理方式を適用するためにアプリケーション２０１に提供する共有データへのアクセス関数の一実施例を示す表である。

【００６９】本実施例の「ｄｓｍ＿」で始まる文字列は、共有データ管理システムを使用するためにアプリケーションが呼び出すオペレーティング・システムの関数の名称である。アプリケーションが共有データにアクセスするためには初期化処理、一貫性保証処理、終了処理を行なう。

【００７０】先ず、１００１及び１００２の初期化処理ｄｓｍ＿ｏｐｅｎ、ｄｓｍ＿ｏｐｅｎ２に於いて、アプリケーションが使用する共有単位を共有データ管理システムに宣言する。具体的にはあるメモリ領域に、共有データ番号を対応付ける。共有データ管理システムでは、共有単位を管理するために必要な管理情報のエントリを確保する。１００１のｄｓｍ＿ｏｐｅｎは、連続したメモリ領域を一つの共有単位として宣言する。１００２のｄｓｍ＿ｏｐｅｎ２は、一定のサイズを持つデータが一定の間隔で並んでいるメモリ領域を一つの共有単位として宣言する。

【００７１】次に１００３及び１００４の一貫性保証処理ｄｓｍ＿ａｃｑｕｉｒｅ、ｄｓｍ＿ｒｅｌｅａｓｅに於いて、アプリケーションが共有単位として宣言したメモリ領域に対してＲｅａｄ／Ｗｒｉｔｅ操作を行なう時は、一連のＲｅａｄ／Ｗｒｉｔｅ操作を１００３のｄｓｍ＿ａｃｑｕｉｒｅと１００４のｄｓｍ＿ｒｅｌｅａｓｅとで囲むことによって、共有データ管理システムは共有単位の一貫性を保証する。

【００７２】最後にアプリケーションが共有データ（共有単位）を必要としなくなった場合には、１００５の終了処理ｄｓｍ＿ｃｌｏｓｅを共有データ管理システムに発行する。この発行によって、初期化処理で確保した共有データ管理システム内の管理情報の領域を解放することができる。

【００７３】尚、初期化処理、一貫性保証処理、終了処理時に共有データ管理システム内で行なわれる処理内容の詳細は後述する。

【００７４】図１２は、本発明による分散共有メモリを利用した領域分割型並列処理を行う際のアプリケーションのメイン処理の一実施例を示すフローチャートである。

【００７５】ここで各計算機上のアプリケーションは、図２及び図５に示した配列データの各要素に対して、先に図２（ｂ）を用いて説明した様な上下左右の隣接要素と自要素の値を用いた演算処理を行う。

【００７６】先ずステップ１１０１で、補助記憶装置などに格納された配列データからアプリケーション２０１（１）に必要な配列データをメインメモリ１０６に読み込む。ステップ１１０２では、アプリケーションで使用する共有データの宣言を行う。具体的には、部分配列５０１のメモリ領域を共有データ番号「０」、部分配列５０２のメモリ領域を共有データ番号「１」として宣言する。図中では、それぞれの宣言を（ｏｐｅｎ　０）、（ｏｐｅｎ　１）と略記したが、これはそれぞれ１００１のルーチンを用いて行う。ステップ１１０３では、アプリケーション２０１（１）が取り扱う部分配列の中で、共有データ番号「０」を持つ共有データを仮想的な共有空間に反映する操作をルーチン１００４を用いて実行する。図中では、（ｒｅｌ　０）と略記した。

【００７７】次のステップ１１０４からステップ１１０８までが、アプリケーションのメインループに対応する。ステップ１１０４で他のアプリケーションと同期を取る。ステップ１１０５では、仮想的な共有空間から最新データを獲得するための操作を実行する。ここでは、移動元を共有データ「１」、移動先を同じ共有データ「１」として１００３のｄｓｍ＿ａｃｑｕｉｒｅ処理を実行する。図中では、この操作を（ａｃｑ　１）と略記した。ステップ１１０６では、部分配列の全ての要素について上下左右の隣接要素と自要素を用いた演算処理を実行する。ステップ１１０７では、ステップ１１０６の結果を仮想的な共有空間に反映する操作をステップ１１０３と同様に実行する。ステップ１１０８では、計算が終了したが否かを判定し、終了していない場合にはステップ１１０４に戻り、ステップ１１０８迄の処理を繰り返し行う。終了した場合にはステップ１１０９に進む。

【００７８】ステップ１１０９では共有データの解放処理を行う。ここでは、共有データ番号「０」を持つ共有データ、共有データ番号「１」を持つ共有データの解放

処理を、ルーチン１００５を用いて行う。図中では、それぞれ（ｃｌｏｓｅ　０）、（ｃｌｏｓｅ　１）と略記した。

【００７９】アプリケーション２０１（２）についても、ステップ１１０１からステップ１１０９と同様な処理を実行する。但し、担当する部分配列が異なるため、ステップ１１１２、ステップ１１１４、ステップ１１１６では、それらに対応するアプリケーション２０１（１）に於ける処理ステップとは異なる共有データに対して操作を行う。

【００８０】図１３は、図１１で示した共有データの初期化処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【００８１】ブロック１２０１で囲まれた処理ステップは、共有データの使用を宣言したアプリケーションを実行している計算機１０１で行なわれる処理内容を、ブロック１２０２で囲まれた処理ステップは、共有データ番号で決定される共有データを管理する計算機で行なわれる処理内容を示している。

【００８２】先ずステップ１２０３で、アプリケーションが共有データ領域管理部４０４に共有データ番号と共有データのメモリ領域を指定して共有データの使用を宣言するために１００１のｄｓｍ＿ｏｐｅｎルーチンを起動する。処理ステップ１２０４では、指定された共有データ番号に基づき、共有データを管理する計算機を決定する。例えば、ネットワーク計算機システムを構成する計算機に予め一意な番号を割り当てておき、共有データ番号をネットワーク計算機システムを構成する計算機の総数で割った余りが、ある予め決定された番号と一致した計算機を管理計算機として決定してもよい。ステップ１２０５では、共有データ領域管理部が決定した共有データの管理計算機に対して共有データを使用することを通知する。　共有データを管理する計算機では、ステップ１２０６に於いて、宣言された共有データがすでに使用されているか否かを確認する。使用されていない場合には、ステップ１２１０で、管理計算機の共有データ管理テーブル４０５に新しいエントリを作成する。また、共有データ保持計算機リストの初期設定を行なう。使用されている場合には、ステップ１２０７に於いて、共有データを使用することを宣言した計算機を６０９の共有データ保持計算機リストに追加し、ステップ１２０８で共有データ保持計算機リストに登録されている計算機に対して共有データ保持計算機リストが変更されたことを通知する。

【００８３】共有データの使用を宣言した計算機では、共有データが宣言されていた場合には、ステップ１２０９で共有データ保持計算機リストに基づき最新の共有データの内容を獲得して、それを共有データの複製とする。宣言されていなかった場合には、ステップ１２１１で共有データの複製のメモリ領域を確保する。

【００８４】ステップ１２１２では、共有データ管理テーブル内に新しいエントリを作成し、共有データアドレスや共有データ複製アドレスなどの初期設定を行なう。

【００８５】図１４は、図１１で示した共有データの一貫性保証処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【００８６】ブロック１３０１で囲まれた処理ステップは、最新の共有データを獲得するために１００３のｄｓｍ＿ａｃｑｕｉｒｅを発行した計算機で行なわれる処理内容を示している。

【００８７】先ずステップ１３０２ではアプリケーション内の共有データに最新の共有データを獲得するために、獲得する共有データ番号（移動元）と獲得した共有データを置くメモリ領域を示す共有データ番号（移動先）を指定して共有データ領域管理部４０４にｄｓｍ＿ａｃｑｕｉｒｅを発行する。

【００８８】ステップ１３０３で獲得したい共有データ番号（移動元）に対応する共有データ複製アドレス、共有データサイズなどを共有データ管理テーブル４０５から共有データ番号をキーとして検索する。

【００８９】次にステップ１３０３と同様にステップ１３０４では、移動先を示す共有データ番号に基づいて、共有データ番号に対応する共有データアドレス、及び共有データサイズを調べる。

【００９０】ステップ１３０５では、移動元である共有データ複製２０５のメモリ領域５１２の内容を移動先の共有データのメモリ領域５１１にコピーを行なう。

【００９１】ステップ１３０５でコピーを行う際には、共有データ管理テーブル４０５の分割フラグ６０６に「１」が格納されている場合、図５に示した領域５１２の格納形式から領域５１１の格納形式への変換を行う。これは、データをコピーする際に、各要素毎に共有データオフセット値６０８分のアドレス間隔でコピーを行うことによって実施する。また同様に、共有データ管理テーブル４０５の構造体定義エントリ６１０に構造体定義が、また階層要素フラグ６１２に「１」が格納されている場合にも、格納形式の変換を行いながら、領域５１２から５１１へのデータコピーを行う。

【００９２】図１５は、図１１で示した共有データの一貫性保証処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【００９３】ブロック１４０１で囲まれた処理ステップは、アプリケーションの共有データを共有データ管理システムに反映するために１００４のｄｓｍ＿ｒｅｌｅａｓｅを発行した計算機で行なわれる処理内容を、ブロック１４０２で囲まれた処理ステップは、共有データの反映があった共有データを保持している全ての計算機で行なわれる処理内容を示している。

【００９４】先ずステップ１４０３ではアプリケーション内の共有データ内容を共有データ管理システムに反映

させるために、共有データ番号を指定して共有データ領域管理部に１００４のｄｓｍ＿ｒｅｌｅａｓｅを発行する。

【００９５】ステップ１４０４では、指定された共有データ番号に対応するメモリ領域を共有データ管理テーブルを共有データ番号をキーとして検索する。更に、共有データ番号付加部が、対応するメモリ領域の内容を読みだし、指定された共有データを保持する全ての計算機に対して変更を伝達するためのパケットを作成する。

【００９６】ステップ１４０４では、共有データのメモリ領域５１１の内容を共有データ複製２０５のメモリ領域５１２にコピーする。この際、図１４のステップ１３０５で行ったコピー処理とは逆方向に処理を実施する。即ち、図５及び図７に示した領域５１１の格納形式から領域５１２の格納形式への変換を行いながら、データコピーを行う。

【００９７】ステップ１４０６では、パケット送受信部が共有データ管理テーブル内の共有データ保持計算機リストに基づいて、共有データ番号付加部が作成したパケットを送信する。

【００９８】共有データを保持している計算機では、ステップ１４０７に於いて、ネットワークによって伝達されてきた共有データの変更を通知するパケットをパケット送受信部で受信する。更にパケット内容を調べて、共有データの変更通知であることを知り、パケット番号識別部にパケットを渡す。

【００９９】ステップ１４０８では、パケット番号識別部がパケット内の共有データ番号を識別する。更にその共有データ番号に基づいて共有データ管理テーブルを検索し、対応する共有メモリ複製アドレス、共有メモリサイズなどを調べる。

【０１００】ステップ１４０９では、共有データ識別部がネットワークから到着した共有データ変更通知に基づいて対応するメモリ領域に対して書き込みを行なう。

【０１０１】図１６は、図１１で示した共有データの終了処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【０１０２】ブロック１５０１で囲まれた処理ステップは、共有データの解放を要求した計算機で行なわれる処理内容を、ブロック１５０２で囲まれた処理ステップは、共有データ番号で決定される共有データを管理する計算機で行なわれる処理内容を示している。

【０１０３】先ず、ステップ１５０３で、アプリケーションが共有データ領域管理部に対して共有データ番号を指定して共有データを解放するために１００４のｄｓｍ＿ｒｅｌｅａｓｅを発行する。処理ステップ１５０４では、指定された共有データ番号に基づき、共有データを管理する計算機を決定する。ステップ１５０５では、共有データ領域管理部が決定した共有データの管理計算機に対して共有データを解放することを通知する。

【０１０４】共有データを管理する計算機では、ステップ１５０６に於いて、解放を宣言された共有データに対して、その共有データ番号に基づいて共有データ管理テーブルの共有データ保持計算機リストから、解放を要求した計算機を削除する。

【０１０５】ステップ１５０７に於いて、共有データ管理テーブルの共有データ保持計算機リストから解放要求のあった計算機を削除した結果、共有データ保持計算機リストが空になったか否かを確認する。計算機リストが空の場合には、ステップ１５０８に於いて、共有データを管理している計算機内の共有データ管理テーブル４０５から共有データのエントリを削除する。計算機リストが空でない場合には、ステップ１５０９に於いて共有データ保持計算機リストに登録されている計算機に対して共有データ保持計算機リストの変更を通知する。

【０１０６】共有データの解放を宣言した計算機では、ステップ１５１０で共有データ管理テーブル内から共有データのエントリを削除し、共有データ複製のために割り当てられていたメモリ領域を解放する。

【０１０７】以上、本発明の実施例について説明してきたが、最後に、再度図４を用いて、本発明の好適な実施例における目的、構成、及び効果について簡単にまとめる。

【０１０８】ネットワークで接続した複数の計算機で構成されるネットワーク計算機システムで、アプリケーションに対してあたかも一つのデータを共有しているかの様にみせる分散共有メモリを実現するために、アプリケーションが使用する共有データに関して、共有するデータ領域や、共有するデーター貫性保証のタイミング等の共有データに関する情報を得て、ネットワーク間を流れるデータ転送量や転送回数を減らし、ネットワーク資源を有効に活用し、共有データへのアクセス性能を向上させる。

【０１０９】本発明の好適な実施例である図４においては、共有データ管理システム２０４は、共有データ番号識別部４０１、共有データ番号付加部４０２、パケット送受信部４０３、共有データ領域管理部４０４から構成され、共有データ管理テーブル４０５を保持する。

【０１１０】アプリケーション２０１が共有するデータ（共有単位）とメモリへの写像を共有データ領域管理部４０４に対して宣言すると、共有データとメモリとの対応を共有データ管理テーブル４０５に格納する。共有データの一貫性を保証するために、共有データ変更を指示した計算機と共有データ変更を受け取る計算機に於いて、次の様な構成を取る。

【０１１１】共有データの変更を指示した計算機では、変更した共有データを他の計算機に通知するために共有データ番号に対応するメモリ領域を調べ、メモリ領域の内容に共有データ番号を付加して、通知用パケットの作成を行なう共有データ番号付加部４０２を設ける。

【０１１２】共有データ変更を受け取る計算機では、ネットワークから伝達されてきたパケットの共有データ番号を識別し、共有データ番号に対応するメモリ領域を共有データ管理部を利用して調べ、対応するメモリ領域に対してデータを書き込む処理を行なう共有データ番号識別部４０１を設ける。

【０１１３】また両計算機に於いてパケットの送受信を行ないネットワーク・インタフェースの制御を行なうパケット送受信部４０３を設ける。

【０１１４】本発明の好適な実施例によれば、アプリケーションが使用する共有データに関する情報を共有データ管理システムに通知する手段を設けたことにより、ネットワークを流れる無駄なデータ転送量や転送回数が減り、ネットワーク資源を有効に利用することができると共に、無駄なデータを転送しないのでネットワークの転送時間も短くなり共有データへのアクセス性能が向上し、例えば領域分割型の並列処理に於て、並列化による高速化率を向上することができる。

【０１１５】

【発明の効果】本発明によれば、ネットワークで接続した複数の計算機システムで構成されるネットワーク計算機システムに於いて、アプリケーションに対してあたかも一つのデータを共有しているかの様にみせる分散共有メモリを実現する際に、共有データの一貫性を保証するために必要なネットワークの転送量や転送回数が減少するので、ネットワーク資源を有効に活用することができる。更に、無駄なデータ転送をしないのでネットワークの転送時間が短くなり、その結果、共有データへのアクセス性能を向上させることができる。

【図面の簡単な説明】

【図１】本発明の一実施例のネットワーク計算機システムのブロック図である。

【図２】本発明の適用対象の一例である領域分割型並列処理の一実施例を示す図である。

【図３】本発明による共有データ保持の一実施例を示す図である。

【図４】本発明に於ける共有データ管理システムの一実施例を示すブロック図である。

【図５】本発明による共有データの領域分割型並列処理への応用を示す一実施例である。

【図６】本発明に於ける共有データ管理テーブルの一実施例を示す図である。

【図７】本発明による共有データのデータ構造体への応用を示す一実施例である。

【図８】本発明に於ける共有データ管理テーブルの一実施例を示す図である。

【図９】本発明に於ける構造体定義テーブルの一実施例を示す図である。

【図１０】本発明に於ける通信パケット構造の一実施例を示す図である。

【図１１】本発明による共有データアクセス関数の一実施例を示す図である。

【図１２】本発明による共有データを利用したアプリケーションのメイン処理の一実施例を示すフローチャートである。

【図１３】本発明による共有データ使用開始処理の一実施例を示すフローチャートである。

【図１４】本発明による共有データ獲得処理の一実施例を示すフローチャートである。

【図１５】本発明による共有データ反映処理の一実施例を示すフローチャートである。

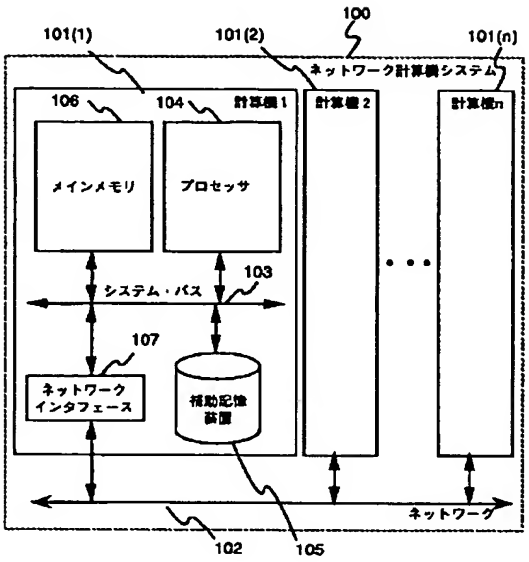
【図１６】本発明による共有データ解放処理の一実施例を示すフローチャートである。

【符号の説明】

１００：ネットワーク計算機システム、１０１：計算機、１０２：ネットワーク、１０３：システム・バス、１０４：プロセッサ、１０５：補助記憶装置、１０６：メインメモリ、１０７：ネットワーク・インタフェース、２０１：アプリケーション、２０２：アプリケーション内のローカルデータ、２０３：アプリケーション内の共有データ、２０４：共有データ管理システム、２０５：共有データ管理システム内の共有データ複製、４０１：共有データ番号識別部、４０２：共有データ番号付加部、４０３：パケット送受信部、４０４：共有データ領域管理部、４０５：共有データ管理テーブル、５０１：共有データ番号「０」を持つ共有データ、５０２：共有データ番号「１」を持つ共有データ、５１１：共有データのメインメモリ上での配置、５１２：共有データ複製のメインメモリ上での配置、６０１：共有データ番号、６０２－６１３：共有データ管理テーブルの管理項目、７０１－７０６：データ構造体の共有データ複製領域上での配置要素、８０１、８０２：構造体定義テーブル、８０３－８０７：構造体定義テーブルの管理項目、９０１：通信パケット、９０２－９０７：通信パケット構成要素、１００１－１００６：共有データのアクセス関数、１３０１－１３１８：共有データを利用した領域分割型アプリケーションのメイン処理ステップ、１２０１：共有データの使用を要求した計算機内での処理ステップ、１２０２：共有データ管理計算機内での処理ステップ、１２０３－１２１２：共有データ使用を要求した時の処理ステップ、１３０１：共有データの獲得処理を発行した計算機内での処理ステップ、１３０２－１３０５：共有データを獲得する時の処理ステップ、１４０１：共有データ反映処理を発行した計算機内での処理ステップ、１４０２：データを共有する他の計算機内での処理ステップ、１４０３－１４１０：共有データを反映する時の処理ステップ、１５０１：共有データ解放を要求した計算機内での処理ステップ、１５０２：共有データ管理計算機内での処理ステップ、１５０３－１５１０：共有データ解放を要求した時の処理ステップ。

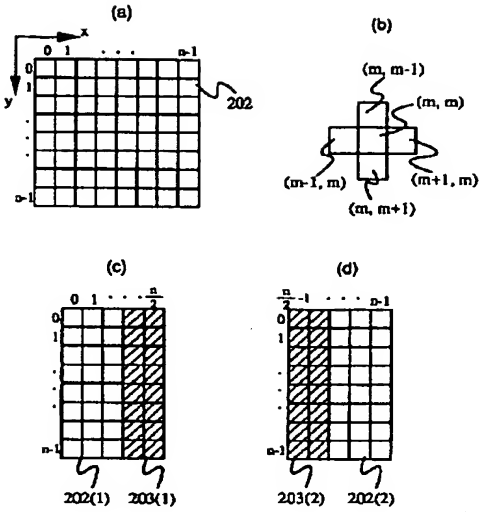
【図１】

図１



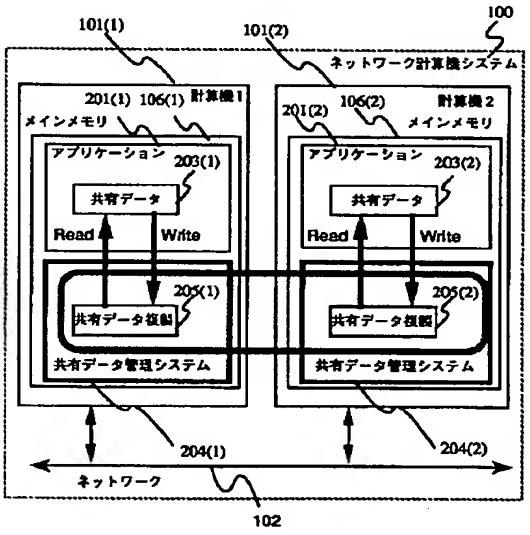
【図２】

図２



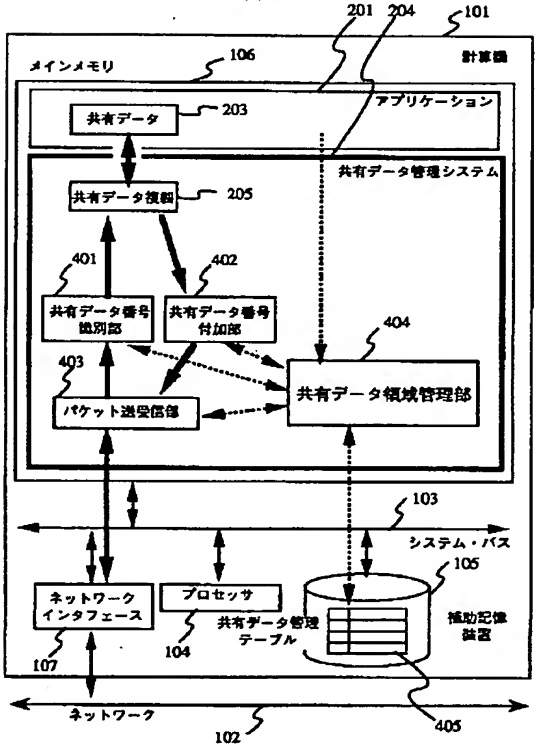
【図３】

図３



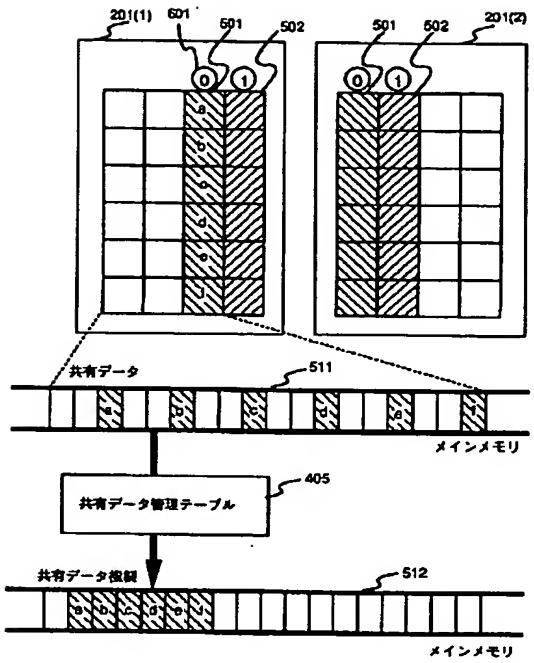
【図４】

図４



【図５】

図5



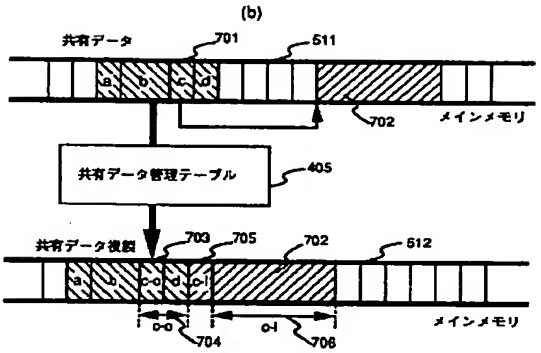
【図７】

図7

(a)

```
struct  ss {
    int     a;
    char    b[8];
    int     *c;
    int     d;
}
```

(b)



【図６】

405

共有データ管理テーブル

共有データ番号	管理ノードフラグ	共有データアドレス	共有データ複製アドレス	共有データサイズ	分割フラグ	共有データ要素数	共有データオフセット	共有データ保持計算機リスト
1	1	101234	1234	2000	0	—	—	1,2,3
5	0	108758	8758	1024	0	—	—	4,5
11	0	104892	4892	8	1	10	80	7,8,9,10
● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ●
6	1	107465	7465	8	1	20	24	1,2,3,4,5,6
● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ● ●	● ●

601　602　603　604　605　606　607　608　609

図8

【図８】

405

共有データ管理テーブル

共有データ番号		共有データ保持計算機リスト	構造体定義	構造体サイズ	階層要素フラグ	階層要素共有データ番号
1		1,2,3	0	0	0	0
5		4,5	0	0	1	6
11	●●●	7,8,9,10	struct ss	40	0	0
●●●		●●●	●●●	●●●	●●●	●●●
6		1,2,3,4,5,6	0	0	0	0
●●●		●●●	●●●	●●●	●●●	●●●

601　602～608　609　610　611　612　613

【図９】

図９

(a)

801

構造体定義テーブル

要素型	要素サイズ	開始位置	データサイズ
int	4	0	
char[8]	8	4	
int *	4	12	16
int	4	16	

803　804　805　806

(b)

802

構造体定義テーブル

要素型	要素サイズ	開始位置	共有データ番号
int	4	0	
char[8]	8	4	
int *	4	12	10
int	4	16	

803　804　805　807

【図１０】

図１０

図８

902　901　903　パケット

パケットヘッダ　データ

904　905　906　907

送信先計算機　送信元計算機　共有データ番号　共有データ管理コマンド

パケットヘッダ

【図１１】

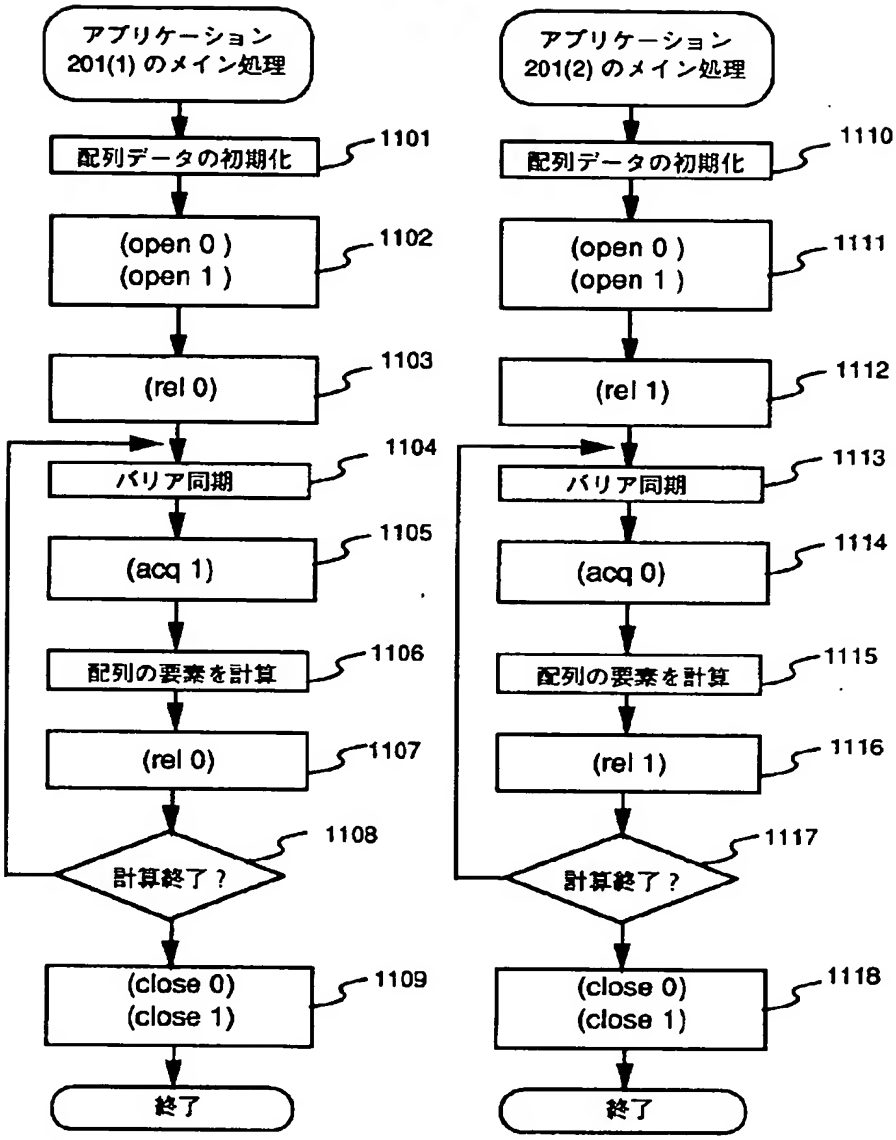
図１１

初期化処理	dsm_open	(共有データ番号、先頭アドレス、領域サイズ)
	dsm_open2	(共有データ番号、先頭アドレス、要素サイズ、要素数、オフセットサイズ)
一貫性保証処理	dsm_acquire	(移動元共有データ番号、移動先共有データ番号)
	dsm_release	(共有データ番号)
終了処理	dsm_close	(共有データ番号)

1001　1002　1003　1004　1005

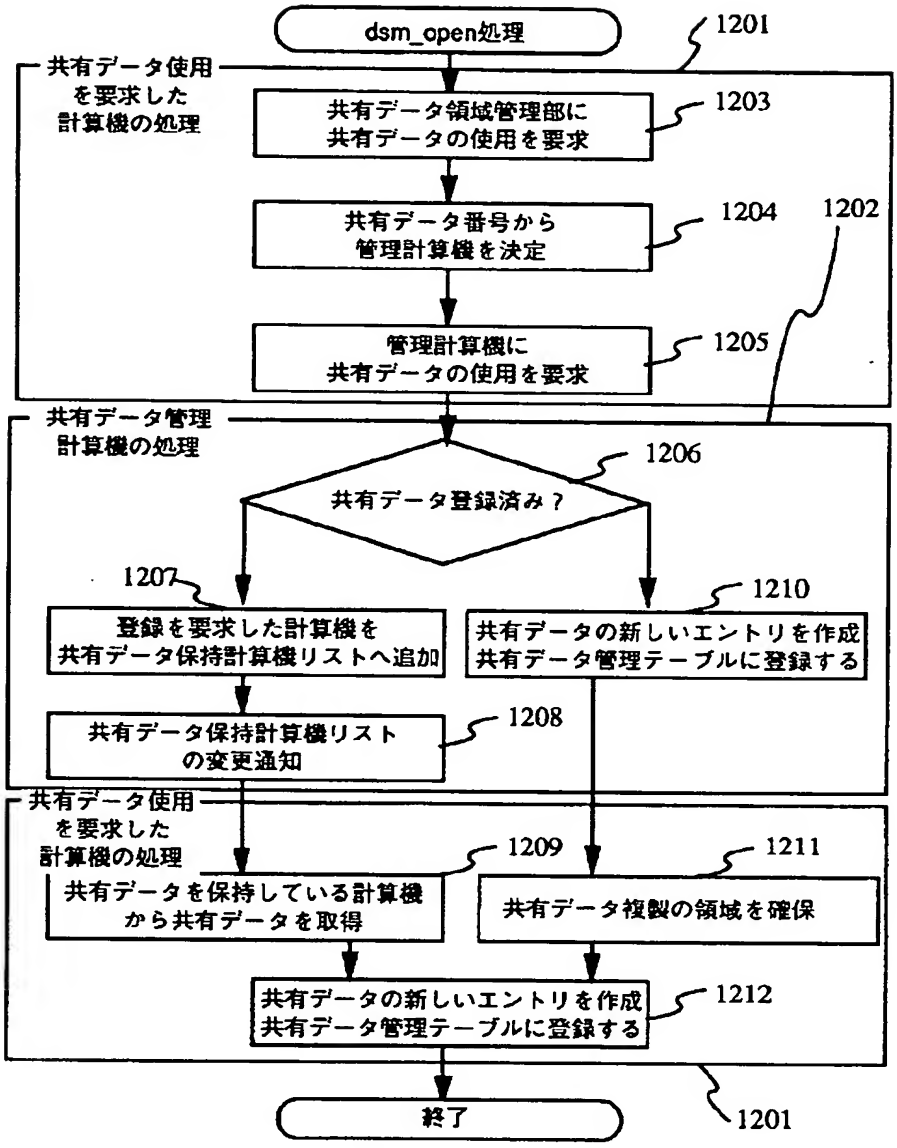
【図１２】

図１２


アプリケーション
201(1) のメイン処理
配列データの初期化
1101
(open 0)
(open 1)
1102
(rel 0)
1103
バリア同期
1104
(acq 1)
1105
配列の要素を計算
1106
(rel 0)
1107
計算終了？
1108
(close 0)
(close 1)
1109
終了
アプリケーション
201(2) のメイン処理
配列データの初期化
1110
(open 0)
(open 1)
1111
(rel 1)
1112
バリア同期
1113
(acq 0)
1114
配列の要素を計算
1115
(rel 1)
1116
計算終了？
1117
(close 0)
(close 1)
1118
終了

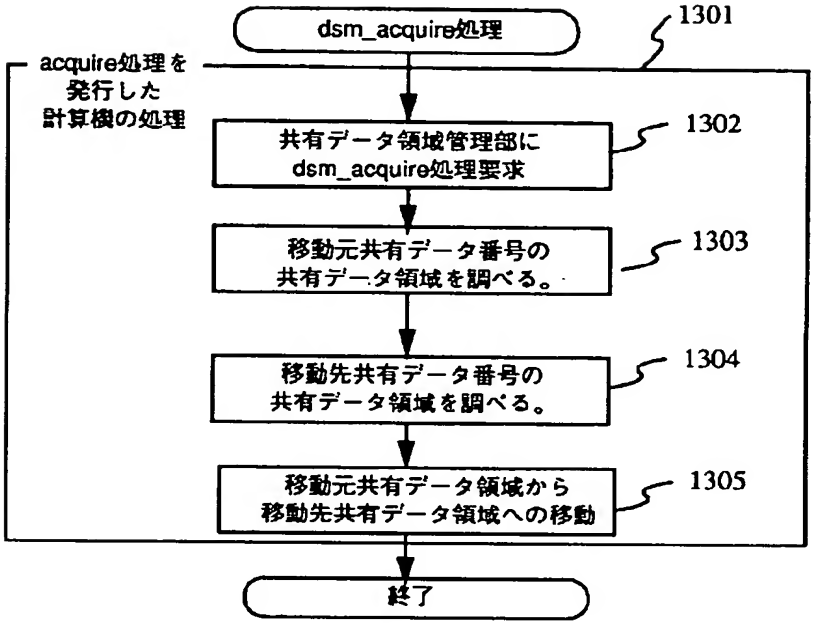
【図１３】

図１３


dsm_open処理
1201
共有データ使用を要求した計算機の処理
共有データ領域管理部に共有データの使用を要求
1203
共有データ番号から管理計算機を決定
1204
1202
管理計算機に共有データの使用を要求
1205
共有データ管理計算機の処理
共有データ登録済み？
1206
1207
登録を要求した計算機を共有データ保持計算機リストへ追加
1210
共有データの新しいエントリを作成共有データ管理テーブルに登録する
共有データ保持計算機リストの変更通知
1208
共有データ使用を要求した計算機の処理
1209
共有データを保持している計算機から共有データを取得
1211
共有データ複製の領域を確保
共有データの新しいエントリを作成共有データ管理テーブルに登録する
1212
終了
1201

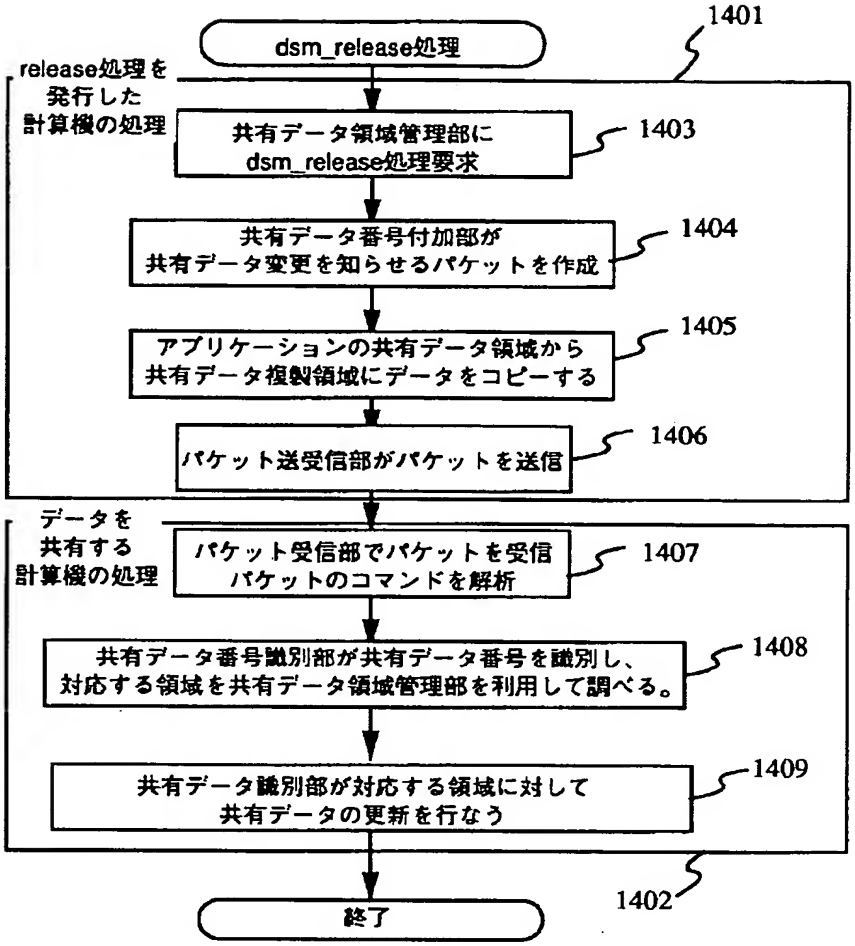
【図１４】

図１４


dsm_acquire処理
1301
acquire処理を
発行した
計算機の処理
共有データ領域管理部に
dsm_acquire処理要求
1302
移動元共有データ番号の
共有データ領域を調べる。
1303
移動先共有データ番号の
共有データ領域を調べる。
1304
移動元共有データ領域から
移動先共有データ領域への移動
1305
終了

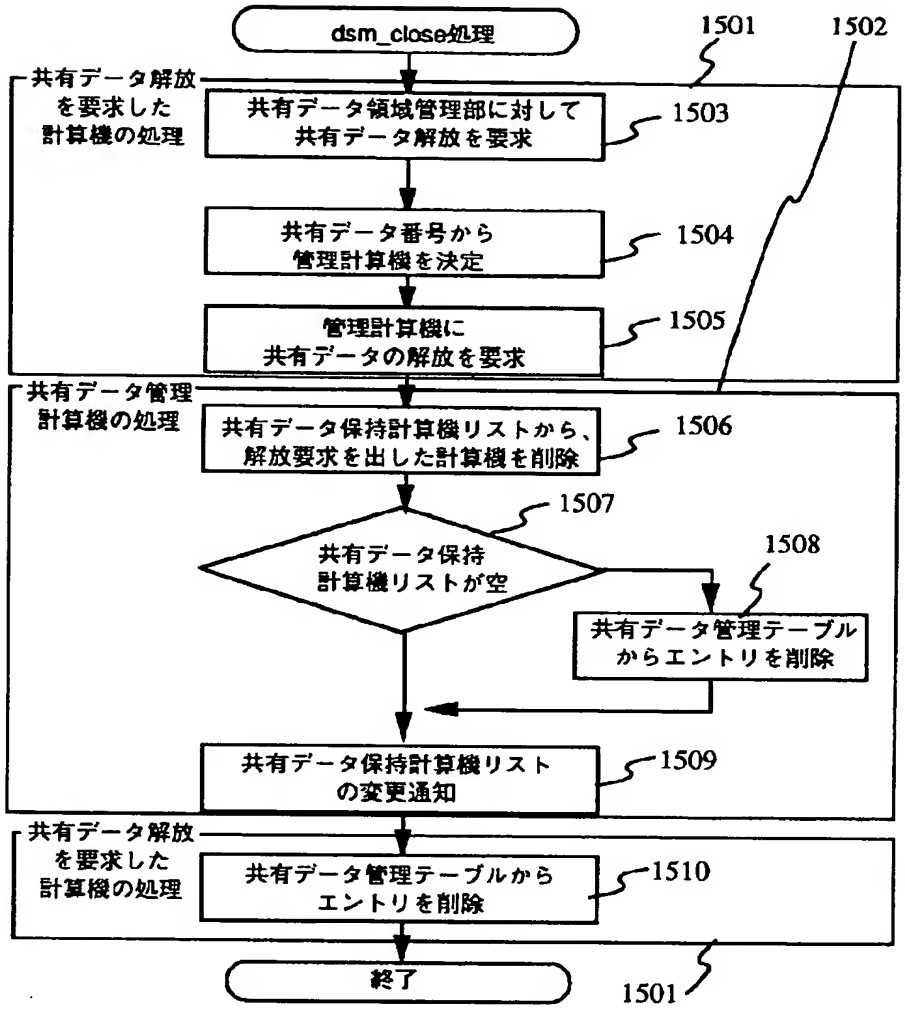
【図１５】

図１５


1401
dsm_release処理
release処理を
発行した
計算機の処理
共有データ領域管理部に
dsm_release処理要求
1403
共有データ番号付加部が
共有データ変更を知らせるパケットを作成
1404
アプリケーションの共有データ領域から
共有データ複製領域にデータをコピーする
1405
パケット送受信部がパケットを送信
1406
データを
共有する
計算機の処理
パケット受信部でパケットを受信
パケットのコマンドを解析
1407
共有データ番号識別部が共有データ番号を識別し、
対応する領域を共有データ領域管理部を利用して調べる。
1408
共有データ識別部が対応する領域に対して
共有データの更新を行なう
1409
1402
終了

【図１６】

図１６



フロントページの続き

(72)発明者　林　剛久
東京都国分寺市東恋ケ窪１丁目280番地
株式会社日立製作所中央研究所内

(72)発明者　鬼頭　昭
神奈川県横浜市戸塚区戸塚町5030番地　株
式会社日立製作所ソフトウェア開発本部内